# 1995 Paper 8 Question 14

**Semantics**

The abstract syntax of IMP commands is given by the following grammar:

$$Com ::= \texttt{skip} \mid Pvar := Iexp \mid Com \ ; Com \mid$$
$$\texttt{if } Bexp \texttt{ then } Com \texttt{ else } Com \mid \texttt{while } Bexp \texttt{ do } Com$$

where $Iexp$ and $Bexp$ are syntactic categories of integer and boolean expressions and $Pvar$ is a set of program variables. Let $States$ be $[Pvar \to \mathbb{Z}]$ and $Cont$, the cpo of *continuations*, be $[States \to A_\perp]$, where $A$ is an unspecified set of program *answers*. A continuation represents what is to be done with the state resulting from the execution of a command in order to return the result of the whole program.

The *continuation semantics* of IMP is defined by giving the meaning $[\![C]\!]$ of each $C \in Com$ as a function which takes a continuation, representing what is to be done when the command has finished, together with a state in which the command is to be executed, and returns an answer:

$$[\![-]\!] \ : \ Com \to (Cont \to (States \to A_\perp)).$$

One clause of the definition of $[\![C]\!]$ is

$$[\![\texttt{skip}]\!] \ k \ S = k(S).$$

Complete the definition of the continuation semantics of IMP commands (expressing their usual behaviour). You may assume that the functions

$$[\![-]\!] \ : \ Iexp \to (States \to \mathbb{Z})$$
$$[\![-]\!] \ : \ Bexp \to (States \to \mathbb{B}) \quad \text{where } \mathbb{B} = \{true, false\}$$

have already been defined. [9 marks]

Now add a new command `abort` to $Com$ and a new error value $Err$ to $A$. The intended behaviour of `abort` is immediately to terminate the entire program, returning $Err$. Extend the continuation semantics of IMP by giving the definition of $[\![\texttt{abort}]\!]$. [2 marks]

Now add two further new command forms:

$$Com \ ::= \ \ldots \mid \texttt{abort} \mid \texttt{exit} \mid Com \texttt{ orelse } Com$$

The intended behaviour of ($C_1$ `orelse` $C_2$) is that it executes exactly like $C_1$ unless $C_1$ hits an `exit` command, in which case further execution of $C_1$ is abandoned and $C_2$ is executed starting in the state at which $C_1$ encountered the `exit`. If $C_1$ does not encounter an `exit` then $C_2$ is ignored. An `exit` command without an enclosing `orelse` behaves like `abort`.

Give a revised continuation semantics to every command of IMP with `abort`, `exit` and `orelse` which reflects this behaviour and in which the denotation of $C \in Com$ is a function which takes *two* continuations and a state and returns an element of $A_\perp$:

$$[\![-]\!] \ : \ Com \to (Cont \to (Cont \to (States \to A_\perp))).$$

Hint: The first continuation is the ordinary default continuation and the second is the continuation to be applied if the command `exit`s. [9 marks]